Terms used **source code commenting** Found **42,966** of **151,219**

Sort results by relevance Save results to a Binder Try an Advanced Search

Display results expanded form Search Tips Try this search in The ACM Guide

Open results in a new window

Results 1 - 20 of 200 Result page: **1** 2 3 4 5 6 7 8 9 10 next

Best 200 shown Relevance scale

1 API documentation from source code comments: a case study of Javadoc

Douglas Kramer

October 1999 **Proceedings of the 17th annual international conference on Computer documentation**

Full text available: pdf(866.34 KB) Additional Information: full citation, abstract, references, citings, index terms

This paper describes in a general way the process we went through to determine the goals, principles, audience, content and style for writing comments in source code for the Java platform at the Java Software division of Sun Microsystems. This includes how the documentation comments evolved to become the home of the Java platform API specification, and the guidelines we developed to make it practical for this document to reside in the same files as the source code.

**Keywords**: API documentation, Java platform, Javadoc, doc comments, doclets, documentation comments, generated documentation, source code comments

2 Poster papers: What's the code?: automatic classification of source code archives

Secil Ugurel, Robert Krovetz, C. Lee Giles

July 2002 **Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining**

Full text available: pdf(759.11 KB) Additional Information: full citation, abstract, references, index terms

There are various source code archives on the World Wide Web. These archives are usually organized by application categories and programming languages. However, manually organizing source code repositories is not a trivial task since they grow rapidly and are very large (on the order of terabytes). We demonstrate machine learning methods for automatic classification of archived source code into eleven application topics and ten programming languages. For topical classification, we concentrate on ...

3 Software and document engineering: Supporting document and data views of source code

Michael L. Collard, Jonathan I. Maletic, Andrian Marcus

November 2002 **Proceedings of the 2002 ACM symposium on Document engineering**

Full text available: pdf(162.30 KB) Additional Information: full citation, abstract, references, index terms

The paper describes the use of an XML format to store and represent program source code.

A new XML application, srcML (SouRCe Markup Language), is presented. srcML presumes a document view of source code where information about the syntactic structure is layered over the original source code document. The resultant multi-layered document has a base layer of all the original text (and formatting). The second layer is the syntactic information, derived from the grammar of the programming language, ...

**Keywords**: XML, abstract syntax tree, markup language, program analysis, source code

4 A model independent source code repository

Anthony Cox, Charles Clarke, Susan Sim

November 1999 **Proceedings of the 1999 conference of the Centre for Advanced Studies on Collaborative research**

Full text available: pdf(157.84 KB) Additional Information: full citation, abstract, references, index terms

Software repositories, used to support program development and maintenance, invariably require an abstract model of the source code. This requirement restricts the repository user to the analyses and queries supported by the data model of the repository. In this work, we present a software repository system based on an existing information retrieval system for structured text. Source code is treated as text, augmented with supplementary syntactic and semantic information. Both the source text an ...

5 Technical papers: design recovery: Browsing and searching source code of applications written using a GUI framework

Amir Michail

May 2002 **Proceedings of the 24th International Conference on Software Engineering**

Full text available: pdf(1.71 MB) Additional Information: full citation, abstract, references, citings, index terms

Nowadays, applications are typically written using an object-oriented GUI framework. In this paper we explore the possibility of using the GUI of such applications to guide browsing and search of their source code. Such a tool would be helpful for software maintenance and reuse, particularly when the application source is unfamiliar. Intuitively, we would expect the task of browsing and searching source code of an application written using a GUI framework to be easier than one that doesn't becau ...

6 Technical papers: design recovery and documentation: Recovering documentation-to-source-code traceability links using latent semantic indexing

Andrian Marcus, Jonathan I. Maletic

May 2003 **Proceedings of the 25th International Conference on Software Engineering**

Full text available: pdf(1.15 MB) Publisher Site Additional Information: full citation, abstract, references, citings, index terms

An information retrieval technique, latent semantic indexing, is used to automatically identify traceability links from system documentation to program source code. The results of two experiments to identify links in existing software systems (i.e., the LEDA library, and Albergate) are presented. These results are compared with other similar type experimental results of traceability link identification using different types of information retrieval techniques. The method presented proves to give ...

7 On enhancing the interface to the source code of computer programs

Ronald Baecker, Aaron Marcus

December 1983 **Proceedings of the SIGCHI conference on Human Factors in Computing Systems**

Full text available: pdf(485.34 KB) Additional Information: full citation, abstract, references, citings, index

terms

This paper addresses issues in the human factors of computer program documentation. We develop a framework for research on enhancing the interface to the source code of computer programs through designing and automating the production of effective typeset representations of the source text. Principles underlying the design research and examples of sample production are presented. This work was supported by the U.S. Defense Advanced Research Projects Ag ...

8 Manipulating source code in DynamicDesign
James Bigelow, Victor Riley
November 1987 **Proceeding of the ACM conference on Hypertext**

Full text available: pdf(924.67 KB) Additional Information: full citation, abstract, references, citings, index terms

DynamicDesign is a Computer-Aided Software Engineering environment for the C language with a layered system architecture for modularity and versatility. DynamicDesign is composed of facilities to edit hypertext objects, maneuver thorough hypertext graphs, build a hypertext graph from a set of existing C source files, and browse source code, documents and system requirements. This paper discusses the DynamicDesign facilities that deal with the source code, sourceBrowser, and source t ...

9 On the inference of configuration structures from source code
Maren Krone, Gregor Snelting
May 1994 **Proceedings of the 16th international conference on Software engineering**

Full text available: pdf(809.64 KB) Additional Information: full citation, references, citings

10 Design principles for the enhanced presentation of computer program source text
R. Baecker, A. Marcus
April 1986 **ACM SIGCHI Bulletin , Proceedings of the SIGCHI conference on Human factors in computing systems**, Volume 17 Issue 4

Full text available: pdf(822.28 KB) Additional Information: full citation, abstract, references, citings, index terms

In order to make computer programs more readable, understandable, appealing, memorable, and maintainable, the presentation of program source text needs to be enhanced over its conventional treatment. Towards this end, we present five basic design principles for enhanced program visualization and a framework for applying these principles to particular programming languages. The framework deals comprehensively with ten fundamental areas that are central to the structure of programming languag ...

11 Object-oriented, single-source, on-line documents that update themselves
Susan Korgen
October 1996 **Proceedings of the 14th annual international conference on Systems documentation: Marshaling new technological forces: building a corporate, academic, and user-oriented triangle**

Full text available: pdf(757.84 KB) Additional Information: full citation, references, citings, index terms

12 ICICLE: groupware for code inspection
L. Brothers, V. Sembugamoorthy, M. Muller
September 1990 **Proceedings of the 1990 ACM conference on Computer-supported cooperative work**

Additional Information:

Full text available: pdf(1.12 MB) full citation, abstract, references, citings, index terms

ICICLE1 ("Intelligent Code Inspection Environment in a C Language Environment") is a multifarious software system intended to augment the process of formal code inspection. It offers assistance in a number of activities, including knowledge-based analysis and annotations of source code, and computer supported cooperative discussion and finalization of inspectors' comments during inspection meetings. This paper reports the implementation of ICICLE and GroupWa ...

13 An interactive source commenter for Prolog programs

David Roach, Hal Berghel, John R. Talburt

September 1990 **ACM SIGDOC Asterisk Journal of Computer Documentation , Proceedings of the 8th annual international conference on Systems documentation**, Volume 14 Issue 4

Full text available: pdf(478.07 KB) Additional Information: full citation, abstract, references, index terms

Prolog meta-circular interpreters, i.e., interpreters for Prolog written in Prolog, perform at least two operations on an object program - they parse it and execute its instructions. There is a useful variant of the meta-circular interpreter, the meta-circular parser, which as its name suggests, parses an object program without executing its instructions. The value of such a parser is that it provides an elegant means to modify Prolog source code. As the object program is parsed, new inform ...

14 Supporting long-term collaboration in software maintenance

Robert Lougher, Tom Rodden

December 1993 **Proceedings of the conference on Organizational computing systems**

Full text available: pdf(2.20 MB) Additional Information: full citation, references, citings, index terms

**Keywords**: annotative collaboration, hypertext, software maintenance, support for long-term collaboration

15 Software evolution: Identifying redundancy in source code using fingerprints

J. Howard Johnson

October 1993 **Proceedings of the 1993 conference of the Centre for Advanced Studies on Collaborative research: software engineering - Volume 1**

Full text available: pdf(800.37 KB) Additional Information: full citation, abstract, references, citings

A prototype implementation of a mechanism that uses fingerprints to identify exact repetitions of text in large program source trees has been built and successfully applied to a legacy source of over 300 megabytes. This prototype system has provided useful information as well as establishing the scalability of the technology. The approach will form the basis of a suite of tools for the visualization and understanding of programs and will complement other approaches currently under investigation.

16 Teaching programming: An experimental analysis of GAME: a generic automated marking environment

Michael Blumenstein, Steven Green, Ann Nguyen, Vallipuram Muthukkumarasamy

June 2004 **Proceedings of the 9th annual SIGCSE conference on Innovation and technology in computer science education**

Full text available: pdf(193.22 KB) Additional Information: full citation, abstract, references, index terms

This paper describes the Generic Automated Marking Environment (GAME) and provides a detailed analysis of its performance in assessing student programming projects and

exercises. GAME has been designed to automatically assess programming assignments written in a variety of languages based on the "structure" of the source code and the correctness of the program's output. Currently, the system is able to mark programs written in Java, C++ and the C language. To use the system, instructors are requ ...

**Keywords**: automatic assessment, generic marking environment

**17** Using literate programming to teach good programming practices

Stephen Shum, Curtis Cook

March 1994 **ACM SIGCSE Bulletin , Proceedings of the twenty-fifth SIGCSE symposium on Computer science education**, Volume 26 Issue 1

Full text available: pdf(533.12 KB) Additional Information: full citation, abstract, references, citings, index terms

The ability to comprehend a program written by other individuals is becoming increasingly important in software development and maintenance. In an attempt to encourage undergraduate Computer Science students to write informative and usable documentation, the literate programming paradigm was incorporated into the teaching of one undergraduate Computer Science course at Augustana College. This paper describes the concept of literate programming, the experience of using literate programming t ...

**18** Token-based scanning of source code for security problems

John Viega, J. T. Bloch, Tadayoshi Kohno, Gary McGraw

August 2002 **ACM Transactions on Information and System Security (TISSEC)**, Volume 5 Issue 3

Full text available: pdf(221.51 KB) Additional Information: full citation, abstract, references, citings, index terms

We describe **ITS4**, a tool for statically scanning C and C++ source code for security vulnerabilities. Compared to other approaches, our scanning technique stakes out a new middle ground between accuracy and efficiency. This method is efficient enough to offer real-time feedback to developers during coding while producing few false negatives. Unlike other techniques, our method is also simple enough to scan C++ code despite the complexities inherent in the language. Using **ITS4**, we fo ...

**Keywords**: Buffer overflows, race conditions, security analysis

**19** System acquisition based on software product assessment

Jean Mayrand, François Coallier

May 1996 **Proceedings of the 18th international conference on Software engineering**

Full text available: pdf(1.11 MB) Publisher Site Additional Information: full citation, abstract, references, citings, index terms

The procurement of complex software product involves many risks. To properly assess and manage those risks, Bell Canada has developed methods and tools that combine process capability assessment with a static analysis based software product assessment. This paper describes the software product assessment process that is part of our risk management approach. The process and the tools used to conduct a product assessment are described. The assessment is in part based on static source code metrics ...

**Keywords**: Bell Canada, complex software product procurement, inspections, process capability assessment, risk management, risk management approach, software product assessment, software quality, software selection, static analysis based software product assessment, static source code metrics, system acquisition

20 Haddock, a Haskell documentation tool

Simon Marlow

October 2002 **Proceedings of the ACM SIGPLAN workshop on Haskell**

Full text available: pdf(94.53 KB) Additional Information: full citation, abstract, references, index terms

This paper describes Haddock, a tool for automatically generating documentation from Haskell source code. Haddock's unique approach to source code annotations provides a useful separation between the implementation of a library and the interface (and hence also the documentation) of that library, so that as far as possible the documentation annotations in the source code do not affect the programmer's freedom over the structure of the implementation. The internal structure and implementation of ...

**Keywords**: API documentation, Haskell, documentation generation, documentation tool, module system, source-code documentation

Results 1 - 20 of 200 Result page: **1** 2 3 4 5 6 7 8 9 10 next

Terms used **comment tree** Found **36,575** of **151,219**

Sort results by relevance
Display results expanded form
Save results to a Binder
Search Tips
Open results in a new window

Try an Advanced Search
Try this search in The ACM Guide

Results 1 - 20 of 200 Result page: **1** 2 3 4 5 6 7 8 9 10 next
Best 200 shown Relevance scale

1 A linear algorithm for copying binary trees using bounded workspace
K. P. Lee
March 1980 **Communications of the ACM**, Volume 23 Issue 3
Full text available: pdf(347.17 KB) Additional Information: full citation, abstract, references, citings

An algorithm to copy a binary tree in linear time using bounded workspace is presented. The algorithm does not modify the original tree at any time. The copy is constructed in such a way that it can be traversed in a read-only fashion even before the copying process is complete, provided one can distinguish between the original and the copy. The traversal can be carried out in parallel with the copying.

**Keywords**: binary trees, copying, tree traversal

2 A comment on the double-chained tree
T. C. Hu
April 1972 **Communications of the ACM**, Volume 15 Issue 4
Full text available: pdf(75.11 KB) Additional Information: full citation, abstract, references, citings

In 1963, Sussenguth [8] suggested that a file should be organized as a double-chained tree for searching and updating. Patt [5] then obtained the optimum double-chained tree under the assumption that no key may prefix another and that all terminal nodes (items of information) have equal probabilities of being searched. Stanfel [6, 7] explored the relation between the double-chained tree and variable length code [3] and solved a special integer programming problem which corresponds to the ca ...

**Keywords**: binary search tree, double-chained tree, file searching

3 Scaling of multicast trees: comments on the Chuang-Sirbu scaling law
Graham Phillips, Scott Shenker, Hongsuda Tangmunarunkit
August 1999 **ACM SIGCOMM Computer Communication Review , Proceedings of the conference on Applications, technologies, architectures, and protocols for computer communication**, Volume 29 Issue 4
Full text available: pdf(988.95 KB) Additional Information: full citation, abstract, references, citings, index terms

One of the many benefits of multicast, when compared to traditional unicast, is that

multicast reduces the overall network load. While the importance of multicast is beyond dispute, there have been surprisingly few attempts to quantify multicast's reduction in overall network load. The only substantial and quantitative effort we are aware of is that of Chuang and Sirbu [3]. They calculate the number of links *L* in a multicast delivery tree connecting a random source to *m* random and ...

4 Picking fruit from the tree of life: comments on taxonomic sampling and quartet methods

Jonathan H. Badger, Paul Kearney

March 2001 **Proceedings of the 2001 ACM symposium on Applied computing**

Full text available: pdf(77.03 KB) Additional Information: full citation, references, index terms

**Keywords**: phylogenetics, quarter methods, taxonomic sampling

5 Comments on optimality of B-trees

Juergen Klonk

January 1983 **ACM SIGMOD Record**, Volume 13 Issue 2

Full text available: pdf(162.74 KB) Additional Information: full citation, references

6 Document querying and transformation: XPath on left and right sides of rules: toward compact XML tree rewriting through node patterns

Jean-Yves Vion-Dury

November 2003 **Proceedings of the 2003 ACM symposium on Document engineering**

Full text available: pdf(224.44 KB) Additional Information: full citation, abstract, references, index terms

XPath [3, 5] is a powerful and quite successful language able to perform complex node selection in trees through compact specifications. As such, it plays a growing role in many areas ranging from schema specifications, designation and transformation languages to XML query languages. Moreover, researchers have proposed elegant and tractable formal semantics [8, 9, 10, 14], fostering various works on mathematical properties and theoretical tools [10, 13, 12, 14].We propose here a novel way to con ...

7 Generation of formatters for context-free languages

Mark van den Brand, Eelco Visser

January 1996 **ACM Transactions on Software Engineering and Methodology (TOSEM)**, Volume 5 Issue 1

Full text available: pdf(2.33 MB) Additional Information: full citation, abstract, references, citings, index terms, review

Good documentation is important for the production of reusable and maintainable software. For the production of accurate documentation it is necessary that the original program text is not copied manually to obtain a typeset version. Apart from being tedious, this will invariably introduce errors. The production of tools that support the production of legible and accurate documentation is a software engineering challenge in itself. We present an algebraic approach to the generation of tools ...

**Keywords**: document preparation, program generators

8 A tree convolution algorithm for the solution of queueing networks

Simon S. Lam, Y. Luke Lien

March 1983 **Communications of the ACM**, Volume 26 Issue 3

Full text available: pdf(3.31 MB) Additional Information: full citation, abstract, references, citings, index terms

A new algorithm called the tree convolution algorithm, for the computation of normalization constants and performance measures of product-form queueing networks, is presented. Compared to existing algorithms, the algorithm is very efficient in the solution of networks with many service centers and many sparse routing chains. (A network is said to have sparse routing chains if the chains visit, on the average, only a small fraction of all centers in the network.) In such a network, s ...

**Keywords**: algorithms, computational algorithms, design, performance, performance evaluation, product-form solution, queueing networks, sparse routing chains, theory, tree convolution algorithm

9 Algorithm 399: Spanning tree

Jouko J. Seppänen

October 1970 **Communications of the ACM**, Volume 13 Issue 10

Full text available: pdf(748.18 KB) Additional Information: full citation, references

**Keywords**: graph, spanning tree, tree

10 Algorithms on Stings, Trees, and Sequences: Computer Science and Computational Biology

Dan Gusfield

December 1997 **ACM SIGACT News**, Volume 28 Issue 4

Full text available: pdf(1.20 MB) Additional Information: full citation

11 Comments on batched searching of sequential and tree-structured files

Marek Piwowarski

June 1985 **ACM Transactions on Database Systems (TODS)**, Volume 10 Issue 2

Full text available: pdf(181.13 KB) Additional Information: full citation, abstract, references, citings, index terms

Exact formulas for the expected cost savings from batching requests against two types of j-ary trees are given. Approximate expressions are also presented.

12 A comment on optimal tree structures

Larry E. Stanfel

October 1969 **Communications of the ACM**, Volume 12 Issue 10

Full text available: pdf(99.26 KB) Additional Information: full citation, abstract, citings, index terms

In Y.N. Patt's paper "Variable Length Tree Structures Having Minimum Average Search Time" [Comm. ACM 12 (Feb. 1969)], the tree structures obtained are not actually optimal with respect to the two-dimensional chaining devised by Sussenguth in his 1963 paper. This note points out that the result can be described as "optimal" only under the constraint—which Patt implicity assumes—that no key be allowed to prefix another.
**Keywords**: double chaining, file searching, information retrieval, tree structures

13 A new representation for linear lists
Leo J. Guibas, Edward M. McCreight, Michael F. Plass, Janet R. Roberts
May 1977 **Proceedings of the ninth annual ACM symposium on Theory of computing**

Full text available: pdf(831.46 KB) Additional Information: full citation, abstract, references, citings, index terms

We present a new data structure for maintaining a set of records in a linear list according to their key values. This data structure has the property that we can keep a number of fingers at points of interest in the key space (e.g., the beginning or the end of the list), so that access and modification in the neighborhood of a finger is very efficient. In the Section 2 we discuss the general structure of our B-tree. Since we propose to search the tree from a leaf ...

14 Source-to-source translation: Ada to Pascal and Pascal to Ada
Paul F. Albrecht, Philip E. Garrison, Susan L. Graham, Robert H. Hyerle, Patricia Ip, Bernd Krieg-Brückner
November 1980 **ACM SIGPLAN Notices , Proceeding of the ACM-SIGPLAN symposium on Ada programming language**, Volume 15 Issue 11

Full text available: pdf(1.43 MB) Additional Information: full citation, abstract, references, citings

An implementation of translators between Ada and Pascal is described. The method used is to define subsets of each language between which there is a straightforward translation and to translate each source program to its respective sublanguage by program transformations. A common internal tree representation is used. The underlying organization of the translators is described, and some of the difficulties we have confronted and solves are discussed.

15 Session 3: A discipline for robustness or storage reduction in binary search trees
J. Ian Munro, Patricio V. Poblete
March 1983 **Proceedings of the 2nd ACM SIGACT-SIGMOD symposium on Principles of database systems**

Full text available: pdf(455.81 KB) Additional Information: full citation, abstract, references

We develop a method of representing binary search trees in an environment in which pointers and other structural information may be "lost" or "maliciously altered". Our fault tolerant representation permits any 2 field changes to be detected and any 1 to be corrected without significantly increasing to storage requirements of the binary tree. The detection and correction procedures applied to the entire tree require $0(n)$ time.Our discipline is also used to represent binary search trees wi ...

16 A note on optimal doubly-chained trees
Steve Kennedy
November 1972 **Communications of the ACM**, Volume 15 Issue 11

Full text available: pdf(148.35 KB) Additional Information: full citation, abstract, references

This note concerns the recent paper by Hu [2] dealing with doubly-chained trees of the type introduced by Sussenguth [9]. In the second part of the paper, Hu deals with the problem of constructing an optimum weighted doubly-chained tree, under the standard assumption that no key is allowed to prefix another, as found, for example, in Patt [6]. He states that for the weights and elements of Figure 1, the optimum doubly-chained tree with all nodes reachable in less than five steps is the one ...

**Keywords**: binary search tree, doubly-chained tree, file searching

17
Generation of Binary Trees from Ballot Sequences

Doron Rotem, Y. L. Varol
July 1978 **Journal of the ACM (JACM)**, Volume 25 Issue 3

Full text available: pdf(472.75 KB) Additional Information: full citation, references, citings, index terms

**18** BURG: fast optimal instruction selection and tree parsing
Christopher W. Fraser, Robert R. Henry, Todd A. Proebsting
April 1992 **ACM SIGPLAN Notices**, Volume 27 Issue 4

Full text available: pdf(536.26 KB) Additional Information: full citation, citings, index terms

**19** Efficient Planarity Testing
John Hopcroft, Robert Tarjan
October 1974 **Journal of the ACM (JACM)**, Volume 21 Issue 4

Full text available: pdf(1.32 MB) Additional Information: full citation, abstract, references, citings, index terms

This paper describes an efficient algorithm to determine whether an arbitrary graph G can be embedded in the plane. The algorithm may be viewed as an iterative version of a method originally proposed by Auslander and Parter and correctly formulated by Goldstein. The algorithm used depth-first search and has O(V) time and space bounds, where V is the number of vertices in G. An ALGOL implementation of the al ...

**20** On the topology of multicast trees
Robert C. Chalmers, Kevin C. Almeroth
February 2003 **IEEE/ACM Transactions on Networking (TON)**, Volume 11 Issue 1

Full text available: pdf(466.02 KB) Additional Information: full citation, abstract, references, citings, index terms

The benefit derived from using multicast is seemingly dependent upon the shape of the distribution tree. In this paper, we attempt to accurately model interdomain multicast trees. We measure a number of key parameters, such as depth, degree frequency, and average degree, for a number of real and synthetic data sets. We find that interdomain multicast trees actually do share a common shape at both the router and autonomous system levels. Furthermore, we develop a characterization of multicast eff ...

**Keywords**: efficiency, modeling, multicast, topology

Results 1 - 20 of 200 Result page: **1** 2 3 4 5 6 7 8 9 10 next